CRITIKON VITAL ANSWERS™

HOST COMMUNICATIONS

COMPACT NIBP MONITOR

DOCUMENT NUMBER: 8555ER

17 March 1997

Author: O Eagles

Rev	R. No.	Checked by Date	Approved by Date	Reason for Change
01	IN294	A Coombes 27 May 1997	G A Thomas 27 May 1997	Original Document
02	R168	A Coombes 5 Dec 1997	G A Thomas 10 Dec 1997	CR756 Add Binnary comms
03	DAR1124			Describe new protocol software in V5.7. Also add new Host Comms features and UI Interface changes

© Copyright in this document belongs to Critikon Ltd. The information contained herein which is confidential belongs to Critikon Ltd. This document is not to be used for any purpose whatsoever without the prior written consent of Critikon Ltd.

DINAMAP™ *Compact* Monitor

Host Communications

Reference Manual

* TRADEMARK

TABLE OF CONTENTS

INTRODUCTION.....5

CONTENTS OF MESSAGES.....6

COMMUNICATIONS ERRORS.....9

MONITOR MODES OF OPERATION.....11

UNIT IDENTIFIERS.....12

PARAMETER IDENTIFIERS AND COMMANDS.....13

NON-INVASIVE BLOOD PRESSURE COMMANDS.....16

PULSE OXIMETRY COMMANDS.....20

PRINTER COMMANDS.....21

HEART/PULSE RATE COMMANDS.....22

TEMPERATURE COMMANDS.....23

THE “STAR” COMMANDS.....26

APPENDIX A. ELECTRICAL SPECIFICATIONS.....29

APPENDIX B. ASCII CHECKSUM CALCULATION.....30

APPENDIX C. NEGATIVE ACKNOWLEDGE CODES.....32

APPENDIX D. MESSAGES WITH CHECKSUMS.....33

APPENDIX E. ALARM FLAG ENCODING.....34

APPENDIX F. BINARY OUTPUT.....35

APPENDIX G. PROTOCOL REVISION NUMBERS.....48

APPENDIX I. 1846 COMPATIBILITY.....49

INTRODUCTION

This publication describes the detailed design of the host communications protocol for the DINAMAP™ *Compact* Model 7200 family of Monitors.

This document describes Host Protocol revision 01

For a further discussion of the host communications protocol revision number, see (APPENDIX G. PROTOCOL REVISION NUMBERS)

To understand what follows, the reader should have a rudimentary understanding of the C programming language. Data structures and small code fragments have been written using C language notation.

The reader must also understand the concepts of bit, character and message. A bit has the usual definition. A character is an ordered set of eight bits. Messages are ordered sets of characters.

The message is the basic indivisible unit of communications between the host computer and the DINAMAP™ *Compact* Monitor. Messages going from the host to the Monitor are called command messages or just commands. Messages going from the Monitor to the host are called response messages or just responses.

CONTENTS OF MESSAGES

Messages are passed back and forth between the host computer and the DINAMAP™ *Compact* Monitor on an asynchronous, serial, duplex communications link. The default data rate is 9600 bits per second and can be changed. See (APPENDIX A. ELECTRICAL SPECIFICATIONS) for a detailed description of this electrical and mechanical interface.

Characters are sent as 1 start bit, followed by 8 data bits (low order bit first), followed by 1 stop bit. ASCII characters occupy the low order 7 bits of an 8 bit data field. The 8th bit is always zero for ASCII characters. In this document, single ASCII characters and strings of ASCII characters are given in double quotes. These character strings all use the Courier type face; e.g., "`This string uses Courier type face`". The carriage return character is represented by the symbol `<CR>`, which also uses the Courier type face.

All commands and most responses are made up from a subset of the ASCII code set. The only ASCII characters used in this protocol are the printable ASCII characters (values of 32 through 126 or ' ' through '~', inclusive) and the carriage return character (value of 13). By special dispensation, the DINAMAP™ *Compact* Monitor ignores the ASCII null character (value of 0).

A valid message is made up of the following sequence of characters:

"x…xss<CR>"

where "x…x" is a string of from 1 to 126 printable ASCII characters that represent the contents of the message and "ss" is the checksum. The checksum is a 12-bit sum embedded in two printable ASCII characters. The method of calculation of the checksum is described in (APPENDIX B. ASCII CHECKSUM CALCULATION).

Command messages from the host computer to the

DINAMAP™ *Compact* Monitor are either normal commands or a negative acknowledge command.

- Normal Commands:

  "upcd…dss<CR>"

where "u" is the unit address, "p" is a single character parameter identifier, "c" is a single character command identifier, "d…d" is a string of from 0 to 123 characters of command data, and "ss" is a two character checksum.

- Negative Acknowledge Command (NACK):

  "u-nss<CR>"

where "u" is the unit address, the ASCII minus sign is the first character of the command string, "n" is a single decimal digit and "ss" is the checksum. See (APPENDIX C. NEGATIVE ACKNOWLEDGE CODES) for the meaning of "n".

Responses from the Monitor to the host computer are either normal responses, negative acknowledge responses, or fatal error responses.

- Normal Response:

  "upcd…dss<CR>"

The normal command and the normal response have the same structure.

- Negative Acknowledge Response (NACK):

  "u-nss<CR>"

The NACK command and the NACK response have the

same structure.

NOTE

In the previous paragraphs, the message strings were explicitly shown to include the two checksum characters and the terminating carriage return character. In the rest of this document, these trailing characters are not shown, but their existence is implied and their presence is required.

COMMUNICATIONS ERRORS

A number of errors can be detected by either the host or the DINAMAP™ *Compact* Monitor:

- Illegal character errors — Character received is not in the range of `' '` to `'~'`,
- Serial input overrun — Host or Monitor can't keep up with the other side. Since this protocol is fully interlocked, this should never happen.
- Message checksum error — Last two characters of the message don't agree with the locally-computed checksum.
- Incorrect message length.
- Invalid parameter identifier — Parameter identifier, `'p'`, is not valid for the Monitor.
- Invalid command identifier — Command identifier, `'c'`, isn't valid for this parameter identifier `'p'`.
- Timeout errors — Either 3 character times has elapsed between non-`<CR>` characters, or 5 seconds have elapsed after sending a command and no response has occurred. This error is only detected by the host.

Each of the previous errors can be either a low-level error or a high-level error. Low-level errors are illegal character, input overrun, and message checksum errors. All other errors are high-level errors. If either a low-level or high-level error is detected by the host or a high level error is detected by the DINAMAP™ *Compact* Monitor, the side that detects the error should send the appropriate NACK message (see (APPENDIX C. NEGATIVE ACKNOWLEDGE CODES). If a low-level error is detected by the Monitor, the Monitor quietly

ignores the error. This behavior should cause a “response message timeout” to occur at the host.

When a NACK message is received by either the host or the DINAMAP ™ *Compact* Monitor, the receiver should re-transmit the last valid message. The Monitor does not impose a retry limit on NACK messages. The host computer should, however, impose whatever limit is appropriate for the particular application. If the Monitor receives a NACK before the first valid message after it was turned on, the Monitor quietly ignores it.

The DINAMAP ™ *Compact* Monitor does not impose any timeout between received characters, but the host computer should. If power is removed from the Monitor before it can finish sending a response, then the host will wait forever for the response to finish. The Monitor never takes more than 2 character times to send the next character of a response message.

The host computer should also impose a limit on the time to wait for the first character of a response. The Monitor never takes more than 5 seconds after receipt of the carriage return character before it starts sending the response. If this timeout occurs, one of three situations has probably occurred;

1. the Monitor has been turned off,
2. the Monitor is on battery power and the batteries have discharged, or
3. a communications error has occurred on the carriage return character terminating the command.

To deal with this third situation, the host should send an illegal character (say, a 127) followed by a carriage return character. This forces any partial command to be rejected and the Monitor to respond with a NACK message.

MONITOR MODES OF OPERATION

The DINAMAP ™ *Compact* Monitor can be in any one of four modes (or states). It's the responsibility of the host to keep track of the state so the host display can clearly indicate when the Monitor is not in normal operate mode. These states are:

- Power-Off Mode. The Monitor is switched off and cannot respond to any command.

- Operate Mode. This is the normal operating mode of the Monitor. The host computer can send commands and receive status messages.

- Service Mode. When the Monitor is in service mode, the host communications protocol still functions. The host software can determine if service mode is active and should make this clear to the user of the host software. Service mode can only be entered from the Monitor front panel (the procedure that puts the Monitor into service mode is described in the service manual). Once in service mode, operate mode can only be restored by entering power-off mode.

- Safe-State Mode. When an internal error is detected, the Monitor enters the “safe state.” The valves are open, the pump is off and the Monitor ignores all command messages. Once in safe-state mode, operate mode can only be restored by entering power-off mode.

If the DINAMAP ™ *Compact* Monitor is in power-off mode, service mode, or safe-state mode, the host software should take care that it's display doesn't lead the user to believe that the Monitor is still in operate-mode.

UNIT IDENTIFIERS

Normal commands start with a unit identifier. The unit identifier can be any character from ' ' to '~'. The DINAMAP™ *Compact* monitor defaults to unit identifier ' '

The DINAMAP™ *Compact* Monitor responds to either of two addresses. One of these addresses is always ' '. The other address can be set from the front panel of the Monitor. In point-to-point applications, host software should use the blank character as the address identifier.

In multi-drop applications, the host software should use the unit identifier character that the user can set by selecting from the "Main Menu/ More/Service/System/Comms" screen on the Monitor. The identifier character must be unique for all Monitors on a single multi-drop line.

PARAMETER IDENTIFIERS AND COMMANDS

Normal commands have the following form:

"upcd…dss<CR>"

where "u" is the unit identifier, "p" is a single character parameter identifier, "c" is a single character command identifier, "d…d" is a string of from 0 to 123 characters of command data, and "ss" is a two character checksum.

Parameter identifiers are assigned as follows:

"N"	Non-invasive blood pressure
"O"	Pulse oximeter
"P"	Printer
"R"	Heart or pulse rate
"T"	Temperature
"*"	DINAMAP™ *Compact* Monitor— Non-parameter-specific

Each parameter identifier has associated with it a set of command identifiers. The command identifiers do not have fixed meaning and must be interpreted in the context of the specified parameter identifier.

The following sections of this document contain detailed descriptions of the commands and responses associated with each parameter identifier. Note that the trailing checksum and carriage return characters have been omitted from these descriptions for clarity only and are still required on the actual commands.

Certain general rules apply to command messages:

- Send commands exactly as shown. Don't send too many or too few characters.

- If a field in a command is numeric, don't send commands with letters or special characters in that field.

- If, due to a host program bug, the above rules are violated, the DINAMAP™ *Compact* Monitor may not gracefully ignore the command. If the checksum is good, the Monitor may process the command and produce unexpected results.

Certain general rules apply to response messages:

- The host should first inspect the status fields in a response message before any data field is displayed or processed. Unless otherwise specified, data fields of response messages are undefined if the status fields of that response message show the parameter to be off, unplugged, in standby mode, or some other unusal condition.

NON-INVASIVE BLOOD PRESSURE COMMANDS

All DINAMAP™ *Compact* Monitor models have one channel of non-invasive blood pressure. The following commands are valid:

Command:	"uNA"	Read non-invasive blood pressure status
Response:	"uNAabcdddeeefffggg"	
	"a"	Determination status "0" - Busy "1" - Done OK "2" - Not used "3" - Determination failed (N99) "4" - Pumpup timeout (N33) "6" - Total time timeout (N44) "7" - One-pressure timeout (N55) "8" - Overpressure (N00) "9" - Unknown NIBP status
	"b"	Adult/neonate status "0" - Unknown "1" - Adult "2" - Neonate
	"c"	Determination type "0" - Normal mode "1" - Stat mode
	"dddd"	Time since last determination of systolic diastolic and mean arterial pressure (in seconds, 0 - 5400)
	"eee"	Systolic pressure
	"fff"	Diastolic pressure
	"ggg"	Mean arterial pressure
Command:	"uNB"	Read cuff pressure channel
Response:	"uNBabbb"	
	"a"	Determination-in-progress indicator "0" - Determination in progress "1" - Determination not in progress
	"bbb"	Current pressure

Command:	"uNCa"	Start a determination
	"a"	Type of determination "0" - Non-stat "1" - Stat mode
Response:	"uNC+" "uNC–"	Determination started OK The determination was not started or network remote control disabled
Command:	"uND"	Cancel a determination
Response:	"uND+" "uND–"	Determination cancelled OK Determination not in progress or network remote control disabled
Command:	"uNE"	Read NIBP alarm limits
Response:	"uNEaaabbbcccdddeeefff"	
	"aaa"	Systolic low limit
	"bbb"	Systolic high limit
	"ccc"	Diastolic low limit
	"ddd"	Diastolic high limit
	"eee"	Mean low limit
	"fff"	Mean high limit

NOTES

The "time since last determination" field in the "uNA" command only counts up to 5,400 seconds (90 minutes). If this field contains a value greater than 5,399, then the determination is too old and the exact age is unknown. In this case, the values in the systolic, diastolic and MAP, adult/neonate and determination type fields are undefined.

All fields of the "uNA" command except the status field become valid *after* a good NIBP determination has completed. During a determination, the status field shows that the determination is in progress and all other fields show

the results of the last good determination (if one existed and isn't too old).

If an in-progress determination completes abnormally, all the fields of the "uNA" command except the status field retain their old values. In this case, the status field shows the reason the determination terminated abnormally.

If an NIBP determination is cancelled, the status goes back to "Done OK." However, the "time since last determination" is not reset. To properly detect the completion of a good determination, the host computer should see the status go from "Busy" to "Done OK" *and* the "time since last determination" go backwards. Note that the host computer may actually miss a time of exactly 0. The host computer should therefore simply look for time to go backwards.

The "uNCa" and "uND" commands can be disabled by the user from the DINAMAP™ *Compact* Monitor front panel. This control is in the SERVICE menu. The path to this screen is:

Main Menu/More/Service/System/Comms

If network control is disabled, these commands respond with a NACK.

PULSE OXIMETRY COMMANDS

The DINAMAP™ *Compact* Monitor has one channel of pulse oximetry. The following commands are valid:

Command:	"uOA"	Read pulse oximetry status
Response:	"uOAabbbcdd"	
	"a"	Channel status "0" - Standby mode "1" - Operate mode OK "3" - No data (P55) "6" - Sensor unplugged
	"bbb"	Oxygen saturation
	"c"	Signal strength ('0' to '9')
	"dd"	Averaging interval

Command:	"uOB"	Read pulse oximetry alarm limits
Response:	"uOBaaabbb"	
	"aaa"	Oxygen saturation low limit
	"bbb"	Oxygen saturation high limit

NOTE : Standby mode (Channel status 0) is only reported when an error of lost signal or lost sensor is acknowledged. Until the error is acknowledged, the DINAMAP™ *Compact* will continue to report a channel status of 3 or 6 as appropriate. Standby mode is only cleared when the DINAMAP™ *Compact* monitor detects a signal at which point the reported channel status will be 1.

PRINTER COMMANDS

The DINAMAP™ *Compact* Monitor has an optional strip printer. The following command is valid:

Command:	"uPS"	Print a snapshot on the printer attached to the Monitor
Response:	"uPS+"	Snapshot started OK
	"uPS–"	Snapshot not started

HEART/PULSE RATE COMMANDS

The heart/pulse rate value displayed on the front panel of the DINAMAP™ *Compact* Monitor can come from one of several sources. The following commands are valid:

Command:	"uRA"	Read heart/pulse rate status
Response:	"uRAabbb"	
	"a"	Heart/pulse rate source "2" - Pulse oximeter "3" - Non-invasive blood pressure
	"bbb"	Heart/pulse rate
Command:	"uRB"	Read heart/pulse rate alarm limits
Response:	"uRBaaabbb"	
	"aaa"	Heart/pulse rate low limit
	"bbb"	Heart/pulse rate high limit

NOTE : The DINAMAP™ *Compact* model T will never report a heart rate source of "2" (pulse oximiter). When pulse oximiter is in standby mode, the heart rate source becomes the NIBP algorithm.

TEMPERATURE COMMANDS

The DINAMAP™ *Compact* Monitor has one channel of temperature. The following commands are valid:

Command:	"uTA"	Read temperature status
Response:	"uTAabbb"	
	"a"	DINAMAP™ *Compact* Monitor display format "1" - Centigrade "2" - Fahrenheit
	"bbbb"	Temperature (in tenths °F) 0 indicates UNPLUGGED
Command:	"uTB"	Read temperature alarm limits
Response:	"uTBaaabbb"	
	"aaaa"	Temperature low limit (in tenths °F)
	"bbbb"	Temperature high limit (in tenths °F)
Command:	"uTC"	
Response:	"uTCabcdddddeeef"	
	"a"	Status "0" - Last determination OK/Predictive temperature idle. "1" - Determination in progress "2" - Determination timeout (E00) "3" - Loss of tissue contact (E11) "4" - Determination abort, probe unplugged (E33)
	"b"	Mode "0" - Normal (predictive) mode "1" - Monitor mode

	"C"	New determination counter. This field contains an ASCII character from ' ' (blank) to '~' (tilde). The ASCII code of the character is incremented each time a determination is started. It "wraps around" from tilde to blank. The starting value is unspecified.

"ddddd"	Time (in seconds) since last determination. If the determination is older than 99990 seconds or the temperature is not available, then this field contains "99999".
"eeee"	Temperature (in tenth degrees F)
"f"	Temperature units used for the bedside display "0" – Degrees F "1" – Degrees C

NOTES

0. **If the mode field indicates predictive mode, then the temperature field of the "uTA" command will show 0s (unplugged).**
1. **The uTB command will return all 0's for upper and lower limits because ther is no alarm limits for predictive temperature.**

THE “STAR” COMMANDS

“Star” commands are used to change communications parameters of the DINAMAP™ *Compact* Monitor or to read instrument status:

Command:	“u*?”	Read model number and host comm. protocol identifier
Response:	“u*?aaaabbc”	
	“aaaa”	Model identifier (7207, 7210, 7216, etc.)
	“bb”	Host communications protocol ID (currently always “01” for this spec)
	“c”	Instrument mode flag “0” — Service mode “1” — Normal operation mode with remote control enabled “2” — Normal operation mode with remote control disabled

Some DINAMAP™ *Compact* Monitor host communications commands control the operation of the Monitor. Two examples are starting and cancelling NIBP determinations. This feature can be disabled by the biomedical engineer from service mode in the Monitor. There is no way to change this or defeat this via the host communications protocol. Once this status has been set in the Monitor, a computer connected via the host communications port can only observe the Monitor, not control it.

Command:	“u*A”	Read DINAMAP™ *Compact* Monitor status
Response:	“u*Atttttttt”	
	“tttttttt”	Time since power-up (in seconds)
Command:	“u*Bn”	Change bit rate

	"n"	Bit rate specifier code "0" — 300 bits per second "1" — 600 bits per second "2" — 1200 bits per second "3" — 2400 bits per second "4" — 4800 bits per second "5" — 9600 bits per second "6" — 19200 bits per second
Response:	"u*B+"	Bit rate changed OK. Note: This response is sent at the new bit rate after a 1-second delay
	"u*B–"	Illegal bit rate specifier code
Command:	"u*C"	Read the counters
Response:	"u*Cabcc"	
	"a"	The good command counter (modulo 64). See the note below
	"b"	The bad command counter (modulo 64). See the note below
	"cc"	The Silence sequence number. See the note below
Command:	"u*L"	Read DINAMAP™ *Compact* Monitor alarm flags. See note below
Response:	"u*Laaaabbbbcccc"	
	"aaaa"	Alarm group 2 (in hex)
	"bbbb"	Alarm group 1 (in hex)
	"cccc"	Alarm group 0 (in hex)

NOTES

The good command counter is incremented (modulo 64) every time a good command is received. The bad command counter is incremented (modulo 64) every time a bad command is received. These counters are each sent as a single printable ASCII character starting with blank (" " or value of 32) through underscore ("_" or value of 95). After the underscore character, the sequence wraps around to blank again.

The silence sequence number follows these rules:

- Every time the SILENCE key is pressed, the silence sequence number changes.
- A number won't repeat until after the SILENCE key has been pressed 100 times.
- If the silence sequence number is less than 50, the alarms at the Monitor are silenced; otherwise, alarms at the Monitor are enabled.

Make no other assumptions.

Alarm flags are sent as three hex groups of 4 digits (16 flags) each. Each group uses the usual encoding of 16 bits into 4 hex digits; i.e., 0x8000 is bit 15 and 0x0001 is bit 0. See (APPENDIX E. ALARM FLAG ENCODING).

APPENDIX A. ELECTRICAL SPECIFICATIONS

The DINAMAP™ *Compact* Monitor has two signal paths, transmit data (output from monitor) and receive data (input to monitor). The electrical levels of these two signals are available in two formats:- RS232C and inverted TTL. The signals from the monitor are NOT isolated from the monitors' functional earth. To connect a host computer to the DINAMAP™ *Compact* family monitor, JJM Ltd. recommends the use of the Isolated Level Converter (JJM Inc. Re-order no. 1925). This device provides electrical isolation providing RS232/E level signals. Refer to Table 1 for complete list of host communication interface signals definitions.

Table 1. Host Communications Interface Signal Definitions

Pin	Signal Description
1	Ground
2	Inverted TTL Transmit Data
3	Inverted TTL Receive Data
4	Fused +5 volts
5	No connection
6	No connection
7	Ground
8	RA (remote alarm)
9	RS232 Request to Send (RTS)
10	RS232 Clear to Send (CTS)
11	RS232 Transmit Data (TxD)
12	No connection
13	RS232 Receive Data (RxD)
14	No connection
15	No connection

APPENDIX B. ASCII CHECKSUM CALCULATION

All messages except the binary block response messages contain an ASCII checksum. The following C function sends a complete ASCII message, including the ASCII checksum.

```
void SendMessage(p)
  char *p;            /* Message content string pointer       */
{
     int sum = 0;     /* The checksum accumulator */

     while(*p)        /* Do the whole string */
     {
            putchar(*p);              /* Send this character */

            sum += (*p++ - ' ' + 1);       /* Sum the ordinal */

            if(sum > 0xFFF)        /* Carry? */
            {
                  sum -= 0xFFF;    /* End-around carry! */
            }

      }

      putchar(' ' + (sum >> 6));         /* High 6 bits 1st     */
      putchar(' ' + (sum & 0x3F));       /* Low  6 bits 2nd     */
      putchar(13);                       /* Carriage return     */
}
```

NOTES

Each printable ASCII character that participates in the checksum is mapped from its normal ASCII representation into an ordinal number from 1 through 95. This is done by “subtracting off a blank and adding one.”

The ordinal numbers of the characters are summed — modulo 2^{12} — with “end-around carry.” End-around carry causes an extra 1 to be added every time there is a carry into the 2^{12} bit position.

The checksum characters themselves do not participate in the checksum calculation.

APPENDIX C. NEGATIVE ACKNOWLEDGE CODES

Errors may occur at several levels within this communications protocol, either by the host or the Monitor. When these are detected, a negative acknowledge (NACK) message is sent. The general form of the negative acknowledge is:

"u-dss<CR>"

where "u" is the unit address, "d" contains an ASCII numeric digit, that identifies the reason for the error and "ss" is the checksum. "d" has standard meaning within this protocol. The standard error codes are:

4	Other message length errors
5	Invalid parameter identifier
6	Invalid command for this parameter identifier
9	Non-specific negative acknowledge

NOTES

The DINAMAP™ *Compact* Monitor responds to all NACK codes by re-transmitting the last valid message. Because of this, the host may group more than one error under a single error code. The limit to this process is for the host to send a non-specific NACK for all errors.

APPENDIX D. MESSAGES WITH CHECKSUMS

Checksums are always present on all commands. During initial host software development, it is convenient to connect the DINAMAP™ *Compact* Monitor to a personal computer and manually communicate with the Monitor via a terminal emulator program. The following table of commands includes the checksum and is handy for reference. Remember to terminate each command string with a carriage return character.

" NA!2"	Read NIBP status
" NB!3"	Read NIBP cuff pressure
" NC0!E"	Start NIBP non-stat
" NC1!F"	Start NIBP stat mode
" ND!5"	Cancel NIBP determination
" NE!6"	Read NIBP alarm limits
" OA!3"	Read pulse oximeter status
" OB!4"	Read pulse oximeter alarm limits
" PS!F"	Generate printer snapshot
" RA!6"	Read pulse/heart rate status
" RB!7"	Read pulse/heart rate alarm limits
" TA!8"	Read temperature status
" TB!9"	Read temperature alarm limits
" TC!:"	Read predictive temperature status
" *? L"	Read model number and host comm. protocol ID
" *A N"	Read DINAMAP™ *Compact* status
" *B2!""	Set host comm. bit rate to 1200 bpm
" *B3!#"	Set host comm. bit rate to 2400 bpm
" *B5!%"	Set host comm. bit rate to 9600 bpm
" *B6!&"	Set host comm. bit rate to 19200 bpm
" *C P"	Read CANCEL count and friends
" *L Y"	Read alarm flags
" *S! "	Go to safe state mode
" -9 I"	Non-specific negative acknowledge
" *Z0000"+"	Stop Binary Mode
" *Z0125"3"	Start binary mode for 10 seconds
" *Z0750"7"	Start binary mode for 1 minute
" *Z9999"O"	Start binary mode indefinately

APPENDIX E. ALARM FLAG ENCODING

Alarm conditions in the DINAMAP™ *Compact* Monitor are encoded as bits in three 16-bit short integers within the Monitor. Each short integer is identified by a group number from 0 to 2. Of the 48 possible alarm flags, there are currently 17 alarm conditions identified in the Monitor:

Alarm Source	**Group #**	**Bit #**
Rate low	0	0
Rate high	0	1
Pulse Ox saturation low	0	2
Pulse Ox saturation high	0	3
NIBP systolic low	0	4
NIBP systolic high	0	5
NIBP diastolic low	0	6
NIBP diastolic high	0	7
NIBP MAP low	0	8
NIBP MAP high	0	9
Temperature high	1	6
Temperature low	1	7
Low battery	1	10
NIBP procedural	1	11
Pulse Ox procedural	1	12
Printer procedural	1	13
Temperature procedural	1	15
RAM lost when powered down	2	1
Reserved	2	2-15

APPENDIX F. BINARY OUTPUT

All commands to the DINAMAP* *Compact* Monitor and most responses from the Monitor are ASCII strings. The only response that is not an ASCII string is the response to the Send Binary Block ("u*Znnnn") command. When the host computer sends this command, the Monitor responds by sending (4 x "nnnn") binary blocks at a rate of 50 blocks per second. This response can take anywhere from 1/50th second to almost 3 minutes and 20 seconds.

BINARY BLOCKS

Each binary block contains 19 bytes. The asynchronous communications parameters are: 8 data bits, no parity bit and 1 stop bit. The structure of the 19 byte binary block is as follows:

```
typedef struct              /* The 50-Hz. binary block        */
{
     char    SeqNum;        /* Block sequence number          */
     char    NonWFData;     /* Non waveform data              */
     char    WFData[15];    /* Waveform data                  */
     char    CSum;          /* Checksum                       */
     char    ocoSeqNum;     /* Ones complement of SeqNum      */

} BHC_BINARY_BLOCK;
```

The following member discussion assumes that an instance of BHC_BINARY_BLOCK has been declared as "xxx".

"SeqNum" — The block sequence number contains an integer that is incremented (modulo 100) on each consecutive block; i.e., 0 to 99 and back to 0. Since 50 blocks are sent per second, "xxx.SeqNum" specifies which block within a 2-second period is being sent. "xxx.SeqNum" is used to detect lost blocks and subsequent loss of sync. Refer to the discussion of block synchronization at the end of this appendix. "xxx.SeqNum" is also used to control data interleaving.

"NonWFData" — This member of the structure contains one char of the 50-byte non-waveform data structure. The character offset into the non-waveform data structure is specified by ("xxx.SeqNum" % 50). Since binary blocks are sent at a rate of 50 blocks per second, the complete non-waveform data structure is assembled once per second. See the description of the non-waveform data structure below.

"WFData" — The waveform data are stored as a 120-bit string. This 120-bit string is stored in the array "xxx.WFData" with "xxx.WFData[0]" holding bits 119 through 112 and "xxx.WFData[14]" holding bits 7 through 0. The high-to-low order of the bits in the 120-byte string is preserved in each byte.

The 120-bit string contains contains four samples from each of three 10-bit waveform channels. These channels are designated channel-0, channel-1 and channel-2. Further, samples for each channel are interleaved across groups of four binary blocks. 16 consecutive samples from each of the three 10-bit channels are embedded in the 120-bit strings of four consecutive binary blocks as follows:

Sample number (modulo 16) for n = xxx.SeqNum % 4					
ChannelNumber	**n is 0**	**n is 1**	**n is 2**	**n is 3**	**Bit Range**
0	0	1	2	3	119 - 110
0	4	5	6	7	109 - 100
0	8	9	10	11	99 - 90
0	12	13	14	15	89 - 80
1	0	1	2	3	79 - 70
1	4	5	6	7	69 - 60
1	8	9	10	11	59 - 50
1	12	13	14	15	49 - 40
2	0	1	2	3	39 - 30
2	4	5	6	7	29 - 20
2	8	9	10	11	19 - 10
2	12	13	14	15	9 - 0

Each 10-bit channel can contain data values from 0 to 1022. A data value of 1023 indicates invalid data.

Channel 0 & Channel 1 are reserved for future use.

Channel-2 is used for plethysmograph and spare waveform data. Plethysmograph waveform data are stored in the even-numbered samples. The spare waveform data are stored in the odd-numbered samples. Plethysmograph and spare waveform data are therefore each available at a rate of 100 10-bit samples per second. Plethysmograph data are uncalibrated. About all that can be said about the plethysmograph waveform is that big data values correspond to systole and small data values correspond to asystole.

"CSum" — The checksum is an 8-bit cyclic redundancy check of the binary block — excluding "xxx.SeqNum" and "xxx.ocoSeqNum". Note that the only members left to include in the checksum are "xxx.NonWFData" and the "xxx.WFData" array. The generating polynomial is:

$$x^8 + x^7 + x^2 + 1$$

The following C code fragment shows how "xxx.CSum" is calculated.

```
unsigned char CSumTbl[] =
{
   0x00,0x91,0x61,0xF0,0xC2,0x53,0xA3,0x32,
   0xC7,0x56,0xA6,0x37,0x05,0x94,0x64,0xF5,
   0xCD,0x5C,0xAC,0x3D,0x0F,0x9E,0x6E,0xFF,
   0x0A,0x9B,0x6B,0xFA,0xC8,0x59,0xA9,0x38,
   0xD9,0x48,0xB8,0x29,0x1B,0x8A,0x7A,0xEB,
   0x1E,0x8F,0x7F,0xEE,0xDC,0x4D,0xBD,0x2C,
   0x14,0x85,0x75,0xE4,0xD6,0x47,0xB7,0x26,
   0xD3,0x42,0xB2,0x23,0x11,0x80,0x70,0xE1,
   0xF1,0x60,0x90,0x01,0x33,0xA2,0x52,0xC3,
   0x36,0xA7,0x57,0xC6,0xF4,0x65,0x95,0x04,
   0x3C,0xAD,0x5D,0xCC,0xFE,0x6F,0x9F,0x0E,
```

```
    0xFB,0x6A,0x9A,0x0B,0x39,0xA8,0x58,0xC9,
    0x28,0xB9,0x49,0xD8,0xEA,0x7B,0x8B,0x1A,
    0xEF,0x7E,0x8E,0x1F,0x2D,0xBC,0x4C,0xDD,
    0xE5,0x74,0x84,0x15,0x27,0xB6,0x46,0xD7,
    0x22,0xB3,0x43,0xD2,0xE0,0x71,0x81,0x10,
    0xA1,0x30,0xC0,0x51,0x63,0xF2,0x02,0x93,
    0x66,0xF7,0x07,0x96,0xA4,0x35,0xC5,0x54,
    0x6C,0xFD,0x0D,0x9C,0xAE,0x3F,0xCF,0x5E,
    0xAB,0x3A,0xCA,0x5B,0x69,0xF8,0x08,0x99,
    0x78,0xE9,0x19,0x88,0xBA,0x2B,0xDB,0x4A,
    0xBF,0x2E,0xDE,0x4F,0x7D,0xEC,0x1C,0x8D,
    0xB5,0x24,0xD4,0x45,0x77,0xE6,0x16,0x87,
    0x72,0xE3,0x13,0x82,0xB0,0x21,0xD1,0x40,
    0x50,0xC1,0x31,0xA0,0x92,0x03,0xF3,0x62,
    0x97,0x06,0xF6,0x67,0x55,0xC4,0x34,0xA5,
    0x9D,0x0C,0xFC,0x6D,0x5F,0xCE,0x3E,0xAF,



    0x5A,0xCB,0x3B,0xAA,0x98,0x09,0xF9,0x68,
    0x89,0x18,0xE8,0x79,0x4B,0xDA,0x2A,0xBB,
    0x4E,0xDF,0x2F,0xBE,0x8C,0x1D,0xED,0x7C,
    0x44,0xD5,0x25,0xB4,0x86,0x17,0xE7,0x76,
    0x83,0x12,0xE2,0x73,0x41,0xD0,0x20,0xB1,
};


#define DATASIZE (sizeof(xxx.NonWFData)+ \
                  sizeof(xxx.WFData[15]))
unsigned char Data[DATASIZE];
unsigned char crc = 0;

CalcCSum()    // Calculate the checksum
{
    int i;

    for(i=0; i<DATASIZE; i+=1)
    {
        crc = CSumTbl[crc ^ Data[i]];
    }
}
```

"ocoSeqNum" — This member is the ones-complement of "xxx.SeqNum". "xxx.ocoSeqNum" is used along with "xxx.SeqNum" to establish and verify block synchronization. See the discussion of block synchronization.

"ocoSeqNum" is calculated as follows:

```
ocoSeqNum = -xxx.SeqNum - 1;
```

This method works on any computer that does twos-complement arithmetic. Note that all modern microprocessors and microcomputers use twos-complement arithmetic.

NON-WAVEFORM DATA

The non-waveform data (a.k.a. derived parameters) are stored in a 50 byte structure. Each binary block contains one of these bytes and (`xxx.SeqNum % 50`) specifies the byte offset into this structure. The non-waveform data structure is defined as follows:

```
typedef struct                          /* The 1-Hz. block                    */
{
          /*****************************/
          /*** Low Speed Data       ***/
          /*****************************/
          char  lsindex;                /* Index (0 to 59)                    */
          char  lsdata;                 /* Data byte                          */

          /*************/
          /*** NIBP  ***/
          /*************/
          short nibp_sys;               /* Systolic (0 to 240)                */
          short nibp_dias;              /* Diastolic (0 to 240)               */
          short nibp_map;               /* MAP (0 to 240)                     */
          short nibp_age;               /* Age of last determination          */
                                        /* in seconds (0 to 5400)             */
          short nibp_target;            /* Target cuff pressure               */
          char  nibp_status;            /* Bits<3:0> -> status                */
                                        /* Bits<5:4> -> cuff type             */
                                        /* Bits<7:6> -> stat mode flag        */

          /*******************************/
          /*** PREDICTIVE TEMPERATURE ***/
          /*******************************/
          char  temp_status;            /* Bits<3:0> ->                       */
                                        /*  0 = Last Determination OK/        */
                                        /*      Predictive temperature idle   */
                                        /*  1 = Determination in Progress     */
                                        /*  2 = Determination Timeout (E00)   */
                                        /*  3 = Loss of tissue contact (E11)  */
                                        /*  4 = Probe disconnected (E33)      */
                                        /* Bit 4 -> 0 = Centigrade,           */
                                        /*          1 = Fahrenheit            */
                                        /*        (displayed at the bedside)  */
                                        /* Bit 5 -> 0 = Predictive mode,      */
                                        /*          1 = Monitor mode          */
                                        /* Bits<7:6> -> Undefined             */
          char  spare_1;                /* Padding byte. (Always zero)        */
          char  temp_counter;           /* ' ' to '~'new determination counter*/
          short temp_age;               /* in seconds (0 to 5400)             */

          char spare_2[9];              /* Spare bytes. Always zero           */
```

```
        /*****************/
        /*** PULSE OX  ***/
        /*****************/
        char  po_sat;               /* O2 saturation (0 to 100)          */
        char  po_quality;           /* Signal quality (0 to 15)          */
        char  po_status;            /* Status                            */


        short temp_dDegF;           /* Deci-degrees F                    */
                                    /* 0 -> unplugged                    */


        /****************************/
        /*** ECG (WHEN AVAILABLE) ***/
        /****************************/
        char  ecg_status;           /* Status                            */
        char  ecg_leadnum;          /* Lead number (1, 2, or 3)          */
        char  ecg_neonate;          /* 0->adult, 1->neonate              */

        /****************************/
        /*** HEART OR PULSE RATE  ***/
        /****************************/
        char  rate_status;          /* Status and source                 */
        short rate_bpm;             /* Beats per minute (0 - 300)        */

        /***********/
        /*** ETC ***/
        /***********/

        short alarmflags[3];        /* Alarm flags                       */

        char  flags;                /* Misc. flags                       */
                                    /* Bit 0 == 1 -> Service mode        */
                                    /* Bit 1 == 1 -> Remote control enable*/

        char  silencestate;         /* Silence-key-pressed state         */

        char  cmdokseqn;            /* Command OK sequence number        */
                                    /* (host to Compact)                 */

        char  cmdngseqn;            /* Command NG sequence number        */
                                    /* (host to Compact)                 */

        unsigned char  csSnapshotCnt;/* Central Sta. Snapshot req counter */
                                    /* 0-15 in lower nibble              */

        char  spare_3;              /* 1 spare padding byte (Always zero) */

} BHC_OPS_DATA;                     /* Once-per-second data */
```

Note that the member definitions that are of type `char` are distributed so that the members of type `short` fall on even

offsets.

LOW SPEED DATA

The following discussion assumes that there is an instance of `BHC_NON_WAVEFORM_DATA` named "`yyy`".

"`yyy.lsindex`" and "`yyy.lsdata`" are used to build a low-speed data structure just as "`xxx.SeqNum`" and "`xxx.NonWFData`" are used to build the `BHC_NON_WAVEFORM_DATA` data structure. The low-speed data structure contains 60 bytes and is therefore available once per minute. "`yyy.lsindex`" counts from 0 to 59 and is used as an index into the low-speed data structure.

The definition of the low-speed data structure follows:

```
typedef struct                                  /* Low speed data block */
{
          /***  Instrument and Protocol ID stuff   ***/
          char  model;                          /* Instrument model number  */
                                                /* 18 -> Compact T          */
                                                /* 19 -> Compact TS         */
          char  hcp_rev;                        /* Host comm. protocol      */
                                                /* revision number          */

          /*******************************************/
          /*** NIBP alarm limits (all 0 to 240 mmhg) ***/
          /*******************************************/
          short nibp_sys_hi;                    /* Systolic high            */
          short nibp_sys_lo;                    /* Systolic low             */
          short nibp_dias_hi;                   /* Diastolic high           */
          short nibp_dias_lo;                   /* Diastolic low            */
          short nibp_map_hi;                    /* MAP high                 */
          short nibp_map_lo;                    /* MAP low                  */

          char spare_1[30];           /* Padding bytes (Always zero)        */

          /*******************************************/
          /*** PULSE OX alarm limits (all 0 to 100%) ***/
          /*******************************************/
          char  po_sat_hi;                      /* O2 sat. high             */
          char  po_sat_lo;                      /* O2 sat. low              */

          char spare_2[4]             /* Padding Bytes. (Always zero)       */

          /*******************************************/
          /*** Heart/Pulse Rate (beats per minute)   ***/
          /*******************************************/
          short rate_bpm_hi;                    /* Rate high (0 to 300)     */
          short rate_bpm_lo;                    /* Rate low  (0 to 300)     */
```

```
        /*****************************************/
        /*** Make this structure be exactly 60 bytes ***/
        /*****************************************/
        char  spare_3[6];                    /* Pad at end (Always zero) */

} BHC_OPM_DATA;                              /* Once-per-minute data */
```

The following discussion assumes that an instance of the `BHC_LOW_SPEED_DATA` structure has been declared and is named "`zzz`".

"`zzz.model`" is used to identify which model of DINAMAP* *COMPACT* Monitor is connected. The data structures and communications protocols are common to all models of the Monitor. However, some members of the data structures do not apply to certain models; e.g., invasive pressure alarm limit members contain undefined values if "`zzz.model`" contains 0. In general, it is the responsibility of the host to make sure that it does not try to use any data value from a parameter that doesn't exist (such as invasive pressure on a model 8700 or model 9700 Monitor).

Alarm limits may have a value of 32767 immediately after powerup. This value should be interpreted as "limits not yet available."

BLOCK SYNCHRONIZATION AND READING BLOCKS

"xxx.ocoSeqNum" and "xxx.SeqNum" are used together to establish and verify block synchronization. Block synchronization is the process of determining which byte in a continuous stream of bytes is the first one in each 19-byte binary block. It needs to be established in either of two circumstances:

- The host software starts up and finds that the Monitor is already sending data. Note that this host software might be in the microprocessor controlling an intelligent modem.
- After a communications error.

Block synchronization is established using the following algorithm:

1. Scan characters received from the DINAMAP* *COMPACT* Monitor until the host sees a byte containing a value from 0 to 99 preceded by a value that would be the the value of "xxx.ocoSeqNum" from the previous block. The value from 0 to 99 is stored in "xxx.SeqNum".
2. Read in the next 18 bytes and fill in the remainder of "xxx".
3. If "xxx.ocoSeqNum" does not contain the ones complement of "xxx.SeqNum", go to step 1.
4. Calculate the block checksum and compare this with "xxx.CSum".
5. If the checksums don't agree, go to step 1.
6. Save this "xxx.SeqNum".

The host has now achieved block synchronization. The first binary block should be discarded and the host can then start

reading binary blocks. Each block is assembled by reading the next 19 bytes. The host should detect lost block synchronization using the following three tests:

- "`xxx.SeqNum`" of the new binary block isn't one greater than the saved "`xxx.SeqNum`" from the previous binary block. Note: This operation of adding one is done modulo 100.
- The locally computed checksum doesn't agree with "`xxx.CSum`".
- "`xxx.ocoSeqNum`" doesn't contain a value that is the ones-complement of "`xxx.SeqNum`".

If any one of these three conditions are met, the host should go back to step 1 of the block synchronization algorithm. Note that the block synchronization algorithm is guaranteed to achieve synchronization within 18 binary blocks.

APPENDIX G. PROTOCOL REVISION NUMBERS

This host communications protocol for the DINAMAP ™ *Compact* Monitor is identified by a unique small integer called the "host comm. protocol revision number." This revision number can be determined in any one of three ways:

- The revision number displayed on the network screen,
- The "`u*?`" host comm. command
- Look at the unit menu screen located at "Main Menu/ More/Service/System/Comms".

This document describes the data structures and commands used to accomplish host communications with a DINAMAP™ *Compact* Monitor. When a new version of the DINAMAP™ *Compact* Monitor software is released, the protocol revision number will not change so long as these data structures and commands do not change. If there is some change in this protocol, then the protocol revision number will change.

NOTE

If the host communication revision number changes, the software revision must also change. If new DINAMAP™ *Compact* Monitor software is released, the monitor software revision must necessarily change, but the host communication revision number may not change.

APPENDIX I. 1846 COMPATIBILITY

TERMS

The three protocols used in the DINAMAP™ *Compact* model 7200 family of Monitors are termed **Standard**, **Binary** and **1846Compatibility**. The **Standard** and **Binary** protocols are described elsewhere in this publication.

The commands implemented for **1846Compatibility** work essentially the same as for 1846SX and 1846SX/OXYTRAK Monitors, except that communications are not limited to the 600 baud rate. The baud rates for *Compact* Monitors are selectable from a service mode menu. The commands implemented for *Compact* family of Monitors are fully described in this appendix.

SELECTING COMMUNICATIONS PROTOCOL

For Monitor software that supports 1846Compatibility mode, the Service Mode Network screen is found by selecting form the “Main menu/More/Service/System/Comms”. This screen displays the following information.

Remote Op.	**Standard**
Address	**1846**
RATE	**Ver. 01**

The available settings and means for each of these items are described in Table I-1.

The Service Mode Network screen software offers the rest of the software global access to determine which protocol is in effect.

The onus is on the operator to ensure that Host and *Compact* Monitor are both set up the same way.

Table I-1. Service Mode Network Screen Settings

Selection	Action
Remote Op.	This allows user to enable or disable the remote operation. ADDRESS no effect RATE no effect STANDARD no effect 1846 no effect. Note: default is disabled.
Address "x"	This allows the user to set up a unique unit address. REMOTE OP. no effect RATE no effect STANDARD no effect 1846 no effect. Note: default is 65 (which is A in ASCII)
RATE xxx	This allows the user to set the default unit baud rate. REMOTE OP. no effect ADDRESS no effect STANDARD no effect 1846 no effect. *Note*: default is 9600 baud
Standard	This allows the user to select standard ASCII host comms protocal. REMOTE OP. no effect RATE set to 9600 ADDRESS no effect 1846 disabled *Note*: If rate other than 9600 is desired, user must change.
1846	This allows the user to select 1846 host comms protocal. REMOTE OP. no effect RATE set to 600 ADDRESS no effect STANDARD disabled *Note*:

	If rate other than 600 is desired, user must change.
Ver xx	This displays the current host comms protocal that the unit is running.

The “ OK “ button returns the user to the top level service menu.

HOST COMMAND FORMAT IN 1846COMPATIBILITY MODE

In 1846Compatibility Mode, command format is

<Modality><*><Command><cr>

where

<Modality> is a single character:

'B' or 'b' for NIBP

'E' or 'e' for Oximeter

<*> is a single character, either the character '*' itself or A-Z or a-z.

<Command> is a single character denoting the command for that modality.

<cr> is a carriage return character.

Examples:

'B*B<cr>' is the command to start an NIBP determination.

'BKB<cr>' is the same as 'B*B<cr>'.

NULL (Hex 00) and LINEFEED (Hex 0A) characters are always ignored and have no effect on the treatment of a command received. Other control characters, or invalid command construction, will cause the whole command to be ignored until the next <cr> character is received, after which a new command may be sent.

Uppercase and lowercase letters are treated the same.

HOST COMMANDS

NIBP commands are described in Table I-2 and oximeter commands are described in Table I-3.

NOTES

1. For historical compatibility reasons, the character (*) stands for either the character (*) itself, or for any character A-Z (or a-z).

2. All commands must be terminated with a carriage return character, Hex 0D.

Table I-2. NIBP Commands

Host Command	Response
B*A	Returns a 30-character string indicating overall NIBP status: **BBAhhmmssMDhhmmssMMMPPPSSSDDD**<cr> where BBA signals this as an NIBP status response hhmmss is the current system time (24-hour clock) M is the current NIBP operating mode '0' for Manual Mode '1' for Auto Mode '2' for STAT Mode D is the NIBP Determination status '0' if no NIBP determination is in progress '1' if an NIBP determination is in progress Note: always '1' while in STAT mode. hhmmss is the time when BP and Pulse were last determined. MMM is the last successful MAP (mmHg). PPP is the last successful Pulse (bpm). SSS is the last successful Systolic (mmHg). DDD is the last successful Diastolic (mmHg). (Note that BP and Pulse values are all updated indivisibly; they are always from the same determination; Pulse may or may not be the one shown on the face of the Monitor, depending on the current heart rate source.)
B*B	Will start a normal NIBP determination. If a determination is in progress, no

	effect. If the Service Mode Network Screen shows Host Control Disabled, then this command has no effect.

Table I-2. NIBP Commands (Continued)

Host Command	Response
B*C	Returns a 30-character string indicating NIBP and its alarm status: **BBCLaaaCCMDhhmmssMMMPPPSSSDDD**<cr> where BBC — signals this as an NIBP alarm + status response L — indicates whether limits have changed recently ‘0’ if limits the same as last time Host asked ‘1’ if limits are different from the last time Host asked aaa — is ‘000’ if no NIBP procedural or system alarm exists else a 3-character code: 002 Determ. aborted 899 Determ. failed 833 Pumpup timeout 844 determ. timeout (too long) 855 Too long at one pressure 800 Overpressure 999 Non-specific error CC — current NIBP cycle time setting, in minutes M — is the current NIBP operating mode ‘0’ for Manual Mode ‘1’ for Auto Mode ‘2’ for STAT Mode D — is the NIBP Determination status ‘0’ if no NIBP determination is in progress ‘1’ if an NIBP determination is in progress Note: always ‘1’ while in STAT mode. hhmmss — is the time when this BP and Pulse were determined. MMM — is the last successful MAP (mmHg). PPP — is the last successful Pulse (bpm). SSS — is the last successful Systolic (mmHg). DDD — is the last successful Diastolic (mmHg). (Note that BP and Pulse values are all updated indivisibly; they are always from the same determination; the Pulse value is as it was determined by the NIBP algorithm, and may or may not be the one shown on the face of the Monitor, depending on the current heart rate display source.)

Table I-2. NIBP Commands (Continued)

Host Command	Response
B*D	Returns a 30-character string with NIBP alarm limits **BBDMMMmmmPPPpppSSSsssDDDddd^^**<cr> where BBD signals this as an NIBP alarm limit response MMMmmm Map Limits high and low (mmHg) PPPppp Pulse Limits high and low (bpm) SSSsss Systolic Limits high and low (mmHg) DDDddd Diastolic Limits high and low (mmHg) ^^ Two spaces for padding.
B*G	Simulates a CANCEL keyhit. If the Service Mode Network Screen shows Host Control Disabled, then this command has no effect.

Table I-3. Oximeter Commands

Host Command	**Response**
E*A	Returns a 30-character string indicating overall Oximeter status: **EAAMhhmmssSSSPPPYYYAA0^^^^^^^**<cr> where EAA signals this as an Oximeter status response M is the current Oximeter operating mode '0' for Standby Mode '1' for Operate Mode hhmmss is the (24-hour clock) time at which SAT and Pulse were determined. SSS Most recent Sat reading (in percent) PPP Most recent Pulse (bpm) YYY Oximeter system alarm code: 000 no alarm, 900 Sensor unplugged 955 No data AA current average interval setting. 0 Alarm limits status (unused, always zero) ^...^ 7 blanks for padding. (Note that the Pulse value is the one determined by the Oximeter algorithm, and it may or may not be the one shown on the face of the Monitor, depending on the current heart rate source.)

USING THE 1846 COMPATIBILITY COMMANDS

1. Simply looking at the “Determination-in-progress” character (which is “0” or “1” to indicate that the Monitor is in WAIT mode or that a determination is in progress, respectively) will not give sufficient information to indicate that new NIBP and Pulse parameters have been displayed on the Monitor’s front panel. This is because some of the alarms will terminate a determination without changing the display of the prior determination results. The state of this character plus noting a change in the “Time of most recent determination” field of the status return block will be sufficient to indicate that new NIBP and Pulse data have been determined.

2. When the monitor is in STAT mode, the character which indicates that a determination is in progress will always be set to “1” when the Host receives a “B*A” or “B*C” response; STAT mode is treated as a single determination lasting up to five minutes. The Host can detect when NIBP and Pulse updates occur by looking at the “Time of determination” field: it will show the time of the most recent update. The so-called systolic update value (sometimes called the systolic predictor) is never transmitted.

3. A Monitor can only process one status request (“B*A”, “B*C” or “B*D”) at a time. The Host should wait for a response to one before issuing another (this is referred to as Half-Duplex).

4. At 600 baud, it takes 1/2 second minimum for the Monitor to transmit a full status block. At higher rates, the transmission is, of course, faster.

5. Host requests may be upper- or lowercase characters (“BBA” is treated the same as “bBa”). Response from the monitor will always be all uppercase and numeric (plus <cr>).

6. NULs and Linefeeds are simply ignored, and may be included in a valid request from the Host. Thus “BBA<cr>“ and “B<NUL>B<LF>A<LF><cr>“ are treated the same.

7. (NIBP)The “B*B” request (to start a determination) is handled exactly as with the **Standard** interface a determination will not start until 30 seconds ofter the last deternination. This is to allow for venous return..

8. Some system failure modes are fatal to the system and preclude any response to a Host request.

9. Before making its first transmission to a Monitor, a Host should transmit a single carriage return character to ensure that any garbage characters are flushed.